

Select

A sophisticated Angular select component (`<aava-select>`) that provides advanced selection capabilities with support for search, multiple selection, icons, form integration, and comprehensive accessibility features.

How to use

Basic Usage

The most basic implementation of the select component with simple options.

```
import { AavaSelectComponent } from "@aava/play-core";
```

Search Functionality

Enable search to filter options dynamically as users type.

```
<aava-select
  label="Select User"
  placeholder="Select a user"
  [showSearch]="false"
>
  <aava-select-option value="john">John Doe</aava-select-option>
  <aava-select-option value="jane">Jane Smith</aava-select-option>
  <aava-select-option value="bob">Bob Johnson</aava-select-option>
  <aava-select-option value="alice">Alice Brown</aava-select-option>
</aava-select>
```

Search Features

- Real-time Filtering : Options filter as you type
- Search Input : Dedicated search field within the select
- No Results State : Clear messaging when no options match
- Keyboard Navigation : Arrow keys work with search results

Multiple Selection

Enable multiple selection mode for selecting multiple options simultaneously.

```

<aava-select
  label="Search Items"
  placeholder="Search and select an item"
  [showSearch]="true">
    <aava-select-option *ngFor="let item of largeDataset" [value]="item.value">
      {{ item.label }}
    </aava-select-option>
  </aava-select>

```

Multiple Selection Features

- Multiple Values : Select multiple options simultaneously
- Visual Distinction : Selected options have different styling
- Comma-separated Display : Multiple selections shown as comma-separated text
- Individual Deselection : Click selected options to deselect them

Icons

Add icons to options and customize the dropdown toggle icon.

```

largeDataset = this.generateLargeDataset(100);

private generateLargeDataset(count: number) {
  const items = [];
  for (let i = 1; i <= count; i++) {
    items.push({
      value: `item-${i}`,
      label: `Item ${i} - Sample Data Entry`
    });
  }
  return items;
}

```

Icon Features

- Option Icons : Add icons to individual options using <aava-icon>
- Custom Dropdown Icon : Customize the dropdown toggle icon
- Icon Integration : Icons work seamlessly with all other features
- Accessibility : Icons properly labeled for screen readers

Sizes

Five size options to fit different layout requirements and visual hierarchies.

```

<!--aava-select
  [multiple]="true"
  label="Select Users"
  [showSearch]="false"
  placeholder="Select users">
  <aava-select-option value="1">John Doe</aava-select-option>
  <aava-select-option value="2">Jane Smith</aava-select-option>
  <aava-select-option value="3">Jack Wilson</aava-select-option>
</aava-select>

```

Available Sizes

- xs (Extra Small) - Extra small size for very compact interfaces
- sm (Small) - Small size for dense interfaces
- md (Medium) - Medium size for most use cases (default)
- lg (Large) - Large size for prominent inputs
- xl (Extra Large) - Extra large size for emphasis and accessibility

States & Validation

Comprehensive validation system with error messages and various input states.

```

<!--aava-select
  label="Countries"
  [showSearch]="false"
  placeholder="Select a country">
  <aava-select-option value="us">
    <aava-icon iconName="circle-check"></aava-icon> United States
  </aava-select-option>
  <aava-select-option value="ca">
    <aava-icon iconName="x-circle"></aava-icon> Canada
  </aava-select-option>
  <aava-select-option value="uk">
    <aava-icon iconName="clock"></aava-icon> United Kingdom
  </aava-select-option>
  <aava-select-option value="de">
    <aava-icon iconName="pause"></aava-icon> Germany
  </aava-select-option>
</aava-select>

<!--aava-select
  label="Custom Dropdown Icon"
  [showSearch]="false"
  placeholder="Select language"
  dropdownIcon="settings">
  <aava-select-option value="javascript">JavaScript</aava-select-option>
  <aava-select-option value="typescript">TypeScript</aava-select-option>
  <aava-select-option value="python">Python</aava-select-option>
</aava-select>

```

Available States

- Normal - Default select state
- Disabled - Non-interactive state with dimmed appearance
- Error - Validation error state with red styling and error message
- Required - Indicates mandatory fields

Form Integration

Comprehensive Angular forms integration with reactive forms and validation.

```
<aava-select size="xs" label="Extra Small" placeholder="Extra Small" [showSearch]="false">
  <aava-select-option value="1">John Doe</aava-select-option>
  <aava-select-option value="2">Jane Smith</aava-select-option>
  <aava-select-option value="3">Jack Wilson</aava-select-option>
  <aava-select-option value="4">James Brown</aava-select-option>
</aava-select>
```

```
<aava-select size="sm" label="Small" placeholder="Small" [showSearch]="false">
  <aava-select-option value="1">John Doe</aava-select-option>
  <aava-select-option value="2">Jane Smith</aava-select-option>
  <aava-select-option value="3">Jack Wilson</aava-select-option>
  <aava-select-option value="4">James Brown</aava-select-option>
</aava-select>
```

```
<aava-select size="md" label="Medium" placeholder="Medium" [showSearch]="false">
  <aava-select-option value="1">John Doe</aava-select-option>
  <aava-select-option value="2">Jane Smith</aava-select-option>
  <aava-select-option value="3">Jack Wilson</aava-select-option>
  <aava-select-option value="4">James Brown</aava-select-option>
</aava-select>
```

```
<aava-select size="lg" label="Large" placeholder="Large" [showSearch]="false">
  <aava-select-option value="1">John Doe</aava-select-option>
  <aava-select-option value="2">Jane Smith</aava-select-option>
  <aava-select-option value="3">Jack Wilson</aava-select-option>
  <aava-select-option value="4">James Brown</aava-select-option>
</aava-select>
```

```
<aava-select size="xl" label="Extra Large" placeholder="Extra Large" [showSearch]="false">
  <aava-select-option value="1">John Doe</aava-select-option>
  <aava-select-option value="2">Jane Smith</aava-select-option>
  <aava-select-option value="3">Jack Wilson</aava-select-option>
  <aava-select-option value="4">James Brown</aava-select-option>
</aava-select>
```

Form Features

- Reactive Forms : Full support for Angular reactive forms

- Template-driven Forms : Works with template-driven forms
- Validation Integration : Seamless integration with Angular validators
- Form Control : Implements ControlValueAccessor interface
- Error Handling : Automatic error message display based on form state

Accessibility

Comprehensive accessibility features ensuring WCAG compliance and inclusive user experience.

Keyboard Navigation

- Toggle Control : Enter/Space : Open/close select Arrow Down : Open select and focus first option Arrow Up : Open select and focus last option Escape : Close select
- Option Navigation : Arrow Up/Down : Navigate between options Enter/Space : Select option Home : Focus first option End : Focus last option Escape : Close select and return to toggle
- Search Navigation : Arrow Down/Up : Navigate from search to options Enter : Select first filtered option Escape : Close select

ARIA Support

- Roles : Proper listbox , option , and combobox roles
- States : aria-expanded , aria-selected , aria-disabled
- Labels : aria-label and aria-labelledby for screen readers
- Live Regions : Dynamic content updates announced to screen readers

API Reference

Inputs

| Property | Type | Default | Description |
|-------------|--------|----------|---|
| placeholder | string | 'Select' | Placeholder text shown when no option is selected |
| label | string | " | Label text displayed above the select |
| id | string | " | Custom element ID |

| Property | Type | Default | Description |
|--------------|---------------------|----------------|---|
| required | boolean | false | Mark field as required for validation |
| dropdownIcon | string | 'chevron-down' | Icon for the dropdown toggle button |
| error | string | " | Custom error message to display |
| disabled | boolean | false | Disable the entire select |
| multiple | boolean | false | Enable multiple selection mode |
| size | SelectboxSize | 'md' | Select size: 'xs' , 'sm' , 'md' , 'lg' , 'xl' |
| showSearch | boolean | true | Enable search functionality |
| inModal | boolean | false | Optimize positioning for modal contexts |
| mode | 'flyout' 'normal' | 'flyout' | Display mode for the dropdown panel |
| expanded | boolean | false | Control expanded state programmatically |
| alwaysOpen | boolean | false | Keep dropdown always open |
| optionOnly | boolean | false | Show only options without toggle |
| parent | boolean | false | Enable hierarchical parent mode |

| Property | Type | Default | Description |
|---------------------|----------------------------|---------|--|
| child | boolean | false | Enable hierarchical child mode |
| hierarchicalOptions | SelectHierarchicalOption[] | [] | Hierarchical options for nested structure |
| customStyles | Record<string, string> | {} | CSS custom properties override |
| allText | string | 'All' | Text for "select all" option in multiple mode |
| allValue | string | 'all' | Value for "select all" option in multiple mode |

Outputs

| Event | Type | Description |
|-----------------|-------------------|--------------------------------|
| selectionChange | EventEmitter<any> | Emitted when selection changes |

SelectboxSize

Methods

The component implements ControlValueAccessor for form integration:

| Method | Parameters | Description |
|-------------------|---------------------|----------------------------|
| writeValue | value: any | Set value programmatically |
| registerOnChange | fn: Function | Register change callback |
| registerOnTouched | fn: Function | Register touched callback |
| setDisabledState | isDisabled: boolean | Set disabled state |

CSS Custom Properties

The select uses CSS custom properties for consistent theming:

| Property | Description |
|--------------------------------|---------------------------------------|
| --select-toggle-padding | Padding inside the toggle button |
| --select-toggle-background | Background color of the toggle |
| --select-toggle-border-radius | Border radius of the toggle |
| --select-toggle-font | Font styling for toggle text |
| --select-brand-color | Brand color for focus and selection |
| --select-brand-text-color | Text color for selected items |
| --select-place-holder-color | Color for placeholder text |
| --select-menu-background | Background color of the dropdown menu |
| --select-menu-border-radius | Border radius of the dropdown menu |
| --select-menu-shadow | Box shadow of the dropdown menu |
| --select-panel-margin | Margin around the dropdown panel |
| --select-panel-max-height | Maximum height of the dropdown panel |
| --select-option-padding | Padding for individual options |
| --select-item-font | Font styling for options |
| --select-item-color | Text color of options |
| --select-item-background | Background color of options |
| --select-item-background-hover | Background color on hover |
| --select-item-text-hover | Text color on hover |
| --select-item-transition | Transition animation for interactions |
| --select-error-color | Color for error messages and state |
| --select-error-border | Border color for error state |
| --select-disabled-text | Text color for disabled state |
| --select-background-disabled | Background color for disabled state |

Best Practices

Implementation Guidelines

- Option Structure : Use meaningful content for option labels
- Performance : For large option lists, consider implementing virtual scrolling
- Search : Enable search for selects with 10+ options
- Icons : Use consistent icon families and sizes
- Validation : Provide clear error messages and validation feedback
- Multiple Selection : Use multiple selection sparingly for better UX

Accessibility Best Practices

- Labels : Always provide descriptive labels for screen readers
- Error Messages : Make error messages specific and actionable
- Keyboard Navigation : Test all keyboard interactions thoroughly
- Focus Management : Ensure focus moves logically through the interface
- ARIA Labels : Use descriptive ARIA labels for complex interactions

Performance Considerations

- Large Lists : Implement virtual scrolling for 100+ options
- Debouncing : Consider debouncing search input for large datasets
- Change Detection : Use OnPush strategy when possible
- Memory : Clean up event listeners and subscriptions properly
- Rendering : Optimize re-rendering with trackBy functions

UX Guidelines

- Option Order : Group related options logically
- Clear Labels : Use clear, concise option names
- Visual Hierarchy : Use consistent spacing and typography
- Feedback : Provide immediate visual feedback for all interactions
- Error States : Make error states clearly visible and understandable
- Search Placeholder : Provide helpful search placeholder text
- Loading States : Show loading indicators for async operations